

Cluster-Aided Mobility Predictions

Jaeseong Jeong[†], Mathieu Leconte[‡] and Alexandre Proutiere[†]

Abstract—Predicting the future location of users in wireless networks has numerous applications, and can help service providers to improve the quality of service perceived by their clients. The location predictors proposed so far estimate the next location of a specific user by inspecting the past individual trajectories of this user. As a consequence, when the training data collected for a given user is limited, the resulting prediction is inaccurate. In this paper, we develop *cluster-aided* predictors that exploit past trajectories collected from *all* users to predict the next location of a given user. These predictors rely on clustering techniques and extract from the training data similarities among the mobility patterns of the various users to improve the prediction accuracy. Specifically, we present CAMP (Cluster-Aided Mobility Predictor), a cluster-aided predictor whose design is based on recent non-parametric bayesian statistical tools. CAMP is robust and adaptive in the sense that it exploits similarities in users' mobility only if such similarities are really present in the training data. We analytically prove the consistency of the predictions provided by CAMP, and investigate its performance using two large-scale datasets. CAMP significantly outperforms existing predictors, and in particular those that only exploit individual past trajectories.

I. INTRODUCTION

Predicting users' mobility in wireless networks has received a great deal of attention recently, strongly motivated by a wide range of applications. Examples of such applications include: location-based services provided to users by anticipating their movements (e.g., mobile advertisement, recommendation systems, risk alarm); urban traffic engineering and forecasting; the design of more efficient radio resource allocation protocols (e.g., scheduling and handover management [1], data prefetching [2] and energy efficient location sensing [3]). However, for these applications to significantly benefit from users' mobility predictions, the latter should be made with a sufficiently high degree of accuracy.

Many mobility prediction methods and algorithms have been devised over the last decade, see e.g. [3]–[6]. The algorithms proposed so far estimate the next location of a specific user by inspecting the data available about her past mobility, i.e., her past trajectory, and exploit the inherent repeated patterns present in this data. These patterns correspond to the regular behavior of the user, e.g. commuting from home to work or visiting favourite restaurants, and need to be extracted from the data to provide accurate predictions. To this aim, one has to observe the behavior of the user over long periods of time. Unfortunately, gathering data about users' mobility can be quite challenging. For instance, detecting the current location of a user with sensors (e.g., GPS, Wi-Fi and cell tower) consumes

a non-negligible energy. Users may also hesitate to log their trajectories to preserve their privacy. In any case, when the data about the mobility of a given user is limited, it is hard to identify her typical mobility patterns, and in turn difficult to provide accurate predictions on her next move or location.

In this paper, we aim at devising mobility predictors that perform well even if the past trajectories gathered for the various users are short. Our main idea is to develop *cluster-aided* predictors that exploit the data (i.e., past trajectories) collected from *all* users to predict the next location of a given user. These predictors rely on clustering techniques and extract from the training data similarities among the mobility patterns of the various users to improve the prediction accuracy. More precisely, we make the following contributions:

- We present CAMP (Cluster-Aided Mobility Predictor), a cluster-aided predictor whose design is based on recent non-parametric bayesian statistical tools [7], [8]. CAMP extracts, from the data, clusters of users with similar mobility processes, and exploit this clustered structure to provide accurate mobility predictions. The use of non-parametric statistical tools allows us to adapt the number of extracted clusters to the training data (this number can actually grow with the data, i.e., with the number of users). This confers to our algorithm a strong robustness, i.e., CAMP exploits similarities in users' mobility only if such similarities are really present in the training data.
- We derive theoretical performance guarantees for the predictions made under the CAMP algorithm. In particular, we show that CAMP can achieve the performance of an optimal predictor (among the set of all predictors) when the number of users grows large, and for a large class of mobility models.
- Finally, we compare the performance of our predictor to that of other existing predictors using two large-scale mobility datasets (corresponding to a Wi-Fi and a cellular network, respectively). CAMP significantly outperforms existing predictors, and in particular those that only exploit individual past trajectories to estimate users' next location.

II. RELATED WORK

Most of existing mobility prediction methods estimate the next location of a specific user by inspecting the past individual trajectories of this user. One of the most popular mobility predictors consists in modelling the user trajectory as an order- k Markov chain. Predictors based on the order- k Markov model are asymptotically optimal [6], [9] for a large class of mobility models. This optimality only holds asymptotically when the length of the observed user past trajectory tends to infinity.

[†]: KTH, Royal Institute of Technology, EE School / ACL, Osquildasv. 10, Stockholm 100-44, Sweden, email: {jaeseong,alepro}@kth.se. [‡]: Huawei, France, email: mathieu.leconte@huawei.com

Unfortunately, when the observed past trajectory of the user is rather short, these predictors perform poorly. Such phenomenon is often referred to as the “cold-start problem”. To improve the performance of these predictors for short histories, a fallback mechanism can be added [4] to reduce the order of the Markov model when the current sequence of k previous locations has not been encountered before. Alternatively, one may adapt the order of the Markov model used for prediction as in the Sampled Pattern Matching (SPM) algorithm [6], which sets the order of the Markov model to a fraction of the longest suffix match in the history. SPM is asymptotically optimal with provable bounds on its rate of convergence, when the trajectory is generated by a stationary mixing source. Another type of mobility predictor, Nextplace [5] attempts to leverage the time-stamps that may be associated with the successive locations visited by the user. Empirical evaluations [3], [4] show that complex mobility models do not perform well: the order-2 Markov predictor with fallback gives comparable performance to that of SPM [6], NextPlace [5] and higher order Markov predictors. In addition [3] reports that the order-1 Markov predictor can actually provide better predictions than higher order Markov predictors, as the latter suffer more from the lack of training data.

There have been a few papers aiming at clustering trajectories or more generally stochastic processes. For example, [10] proposes algorithms to find clusters of trajectories based on likelihood maximization for an underlying hidden Markov model. For the same problem, [11] uses spectral clustering in a semi-parametric manner based on Bhattacharyya affinity metric between pairs of trajectories. Those methods would not work well in our setting. This is due to the facts that (i) users belonging to a same cluster should have trajectories generated by identical parameters, and (ii) the number of clusters should be known beforehand, or estimated in a reliable way. The non-parametric Bayesian approach developed in this paper addresses both issues. [12] also introduced Bayesian approach that focused on the similarity between users’ temporal patterns. But they do not consider the similarity between spatial trajectories and the correlation to the recent locations which are crucial to the correct predictions in our setting.

III. MODELS AND OBJECTIVES

In this section, we first describe the data on past user trajectories available at a given time to build predictors. We then provide a model for user mobility, used to define our non-parametric inference approach, as well as its objectives.

A. Collected Data

We consider the problem of predicting at a given time the mobility, i.e., the next position of users based on observations about past users’ trajectories. These observations are collected and stored on a server. The set of users is denoted by \mathcal{U} , and users are all moving within a common finite set \mathcal{L} of L locations. The trajectory collected for user u is denoted by $x^u = (x_1^u, \dots, x_{n^u}^u)$, where x_t^u corresponds to the t -th location visited by user u , and where n^u refers to the length

of the trajectory. $x_{n^u}^u$ denotes the current location of user u . By definition, we impose $x_t^u \neq x_{t+1}^u$, i.e., two consecutive locations on a trajectory must be different. Let $x^{\mathcal{U}} = (x^u)_{u \in \mathcal{U}}$ denote the set of user trajectories. Observe that the lengths of the trajectories may vary across users. If the location of a user is sensed periodically, we can collect the time a given user has stayed at each location. Those staying times for user u are denoted by $s^u = (s_1^u, \dots, s_{n^u-1}^u)$, where s_t^u is the staying time at the t -th visited location. To simplify the presentation, we present our prediction methods ignoring the staying times s^u ; but we mention how to extend our approach to include staying times in §IV-B4.

Next we introduce additional notations. We denote by $n_{i,j}^u$ the number of observed transitions for user u from location i to j , (i.e., $n_{i,j}^u = \sum_{t=1}^{n^u-1} \mathbb{1}(x_t^u = i, x_{t+1}^u = j)$). Similarly, $n_i^u = \sum_{t=1}^{n^u} \mathbb{1}(x_t^u = i)$ is the number of times user u has been observed at location i . Let $\mathcal{H} \subseteq \bigcup_{n=0}^{\infty} \mathcal{L}^n$ denote the set of all possible trajectories of a given user, and let $\mathcal{H}^{\mathcal{U}}$ be the set of all possible set of trajectories of users in \mathcal{U} .

B. Mobility Models

The design of our predictors is based on a simple mobility model. We assume that user trajectories are order-1 Markov chains, with arbitrary initial state or location. More precisely, user- u ’s trajectory is generated by the transition kernel $\theta^u = (\theta_{i,j}^u)_{i,j \in \mathcal{L}} \in [0, 1]^{L \times L}$, where $\theta_{i,j}^u$ denotes the probability that user u moves from location i to j along her trajectory. Hence, given her initial position x_1^u , the probability of observing trajectory x^u is $P_{\theta^u}(x^u) := \prod_{t=1}^{n^u-1} \theta_{x_t^u, x_{t+1}^u}^u$. Our mobility model can be readily extended to order- k Markov chains. However, as observed in [3], order-1 Markov chain model already provides reasonably accurate predictions in practice, and higher-order models would require a fall-back mechanism¹ [4]. Throughout the paper, we use uppercase letters to represent random variables and the corresponding lowercase letters for their realizations, e.g. X^u (resp. x^u) denotes the random (resp. realization of) trajectory of user u .

C. Bayesian Framework, Clusters, and Objectives

We adopt a Bayesian framework, and assume that the transition kernels of the various users are drawn independently from the same distribution $\mu \in \mathcal{P}(\Theta)$ ² referred to as the *prior* distribution over the set of all possible transition kernels Θ . This assumption is justified by De Finetti’s theorem (see [13], Theorem 11.10) if $(\theta^u)_{u \in \mathcal{U}}$ are exchangeable (which is typically the case if users are a priori indistinguishable). In the following, the expectation and probability under μ are denoted by \mathbb{E} and \mathbb{P} , respectively. To summarize, the trajectories of users are generated using the following hierarchical model: for

¹To accurately predict the next position of user u given that the sequence of her past k positions is i_1, \dots, i_k , her trajectory should contain numerous instances of this sequence, which typically does not occur if the observed trajectory is short – and this is precisely the case we are interested in.

² $\mathcal{P}(\mathcal{M})$ denotes the set of distributions over the set \mathcal{M} , and $\Theta = \{\theta \in [0, 1]^{L \times L} : \forall i, \sum_j \theta_{ij} = 1\}$.

all $u \in \mathcal{U}$, $\theta_u \sim \mu$, $X^u \sim P_{\theta^u}$, and n^u, X_1^u are arbitrarily fixed.

To provide accurate predictions even if observed trajectories are rather short, we leverage similarities among user mobility patterns. It seems reasonable to think that the trajectories of some users are generated through similar transition kernels. In other words, the distribution μ might exhibit a clustered structure, putting mass around a few typical transition kernels. Our predictors will identify these clusters, and exploit this structure, i.e., to predict the next location of a user u , we shall leverage the observed trajectories of all users who belong to user- u 's cluster.

For any user u , we aim at proposing an accurate predictor $\hat{x}^u \in \mathcal{L}$ of her next location, given the observed trajectories $X^{\mathcal{U}} = x^{\mathcal{U}}$ of all users. The (Bayesian) accuracy of a predictor \hat{x}^u for user u , denoted by $\pi^u(\hat{x}^u)$, is defined as $\pi^u(\hat{x}^u) := \mathbb{P}(X_{n^u+1}^u = \hat{x}^u | x^{\mathcal{U}}) = \mathbb{E}[\theta_{x_{n^u+1}^u, \hat{x}^u}^u | x^{\mathcal{U}}]$ (where for conciseness, we write $\mathbb{P}(\cdot | x^{\mathcal{U}}) = \mathbb{P}(\cdot | X^{\mathcal{U}} = x^{\mathcal{U}})$). Clearly, given $X^{\mathcal{U}} = x^{\mathcal{U}}$, the best possible predictor would be:

$$\hat{x}^u \in \arg \max_{j \in \mathcal{L}} \mathbb{E}[\theta_{x_{n^u+1}^u, j}^u | x^{\mathcal{U}}]. \quad (1)$$

Computing this optimal predictor, referred to as the Bayesian predictor with prior μ , requires the knowledge of μ . Indeed:

$$\mathbb{E}[\theta_{i,j}^u | x^{\mathcal{U}}] = \frac{\int_{\theta} \theta_{i,j} P_{\theta}(x^u) \mu(d\theta)}{\int_{\theta} P_{\theta}(x^u) \mu(d\theta)}. \quad (2)$$

Since here the prior distribution μ is unknown, we will first estimate μ from the data, and then construct our predictor according to (1)-(2).

IV. BAYESIAN NON-PARAMETRIC INFERENCE

In view of the model described in the previous section, we can devise an accurate mobility predictor if we are able to provide a good approximation of the prior distribution μ on the transition kernels dictating the mobility of the various users. If μ concentrates its mass around a few typical kernels that would in turn define clusters of users (i.e., users with similar mobility patterns), we would like to devise an inference method identifying these clusters. On the other hand, our inference method should not discover clusters if there are none, nor specify in advance the number of clusters (as in the traditional mixture modelling approach). Towards these objectives, we apply a Bayesian non-parametric approach that estimates how many clusters are needed to model the observed data and also allows the number of clusters to grow with the size of the data. In Bayesian non-parametric approaches, the complexity of the model (here the number of clusters) is part of the posterior distribution, and is allowed to grow with the data, which confers flexibility and robustness to these approaches. In the remaining of this section, we first present an overview of the Dirichlet Process mixture model, a particular Bayesian non-parametric model, and then apply this model to the design of CAMP (Cluster-Aided Mobility Predictor), a robust and flexible prediction algorithm that efficiently exploits similarities in users' mobility, if any exist.

A. Dirichlet Process Mixture Model

When applying Bayesian non-parametric inference techniques [7] to our prediction problem, we add one level of randomness. More precisely, we approximate the prior distribution μ on the transition kernels θ^u by a random variable $\hat{\mu}$ with distribution $g \in \mathcal{P}(\mathcal{P}(\Theta))$. This additional level of randomness allows us to introduce some flexibility in the number of clusters present in μ . We shall compute the posterior distribution g given the observations $x^{\mathcal{U}}$, and hope that this posterior distribution, denoted as $g|x^{\mathcal{U}}$, will concentrate its mass around the true prior distribution μ . To evaluate $g|x^{\mathcal{U}}$, we use Gibbs sampling techniques (see Section IV-B1), and from these samples, we shall estimate the true prior μ , and derive our predictor by replacing μ by its estimate in (1)-(2).

For the higher-level distribution g , we use the Dirichlet Process (DP) mixture model, a standard choice of prior over infinite dimensional spaces, such as $\mathcal{P}(\Theta)$. The DP mixture model has a possibly infinite number of mixture components or clusters, and is defined by a *concentration* parameter $\alpha > 0$, which impacts the number of clusters, and a *base* distribution $G_0 \in \mathcal{P}(\Theta)$, from which new clusters are drawn. The DP mixture model with parameters α and G_0 is denoted by $DP(\alpha, G_0)$ and defined as follows. If ν is a random measure drawn from $DP(\alpha, G_0)$ (i.e., $\nu \sim DP(\alpha, G_0)$), and $\{A_1, A_2, \dots, A_K\}$ is a (measurable) partition of Θ , then $(\nu(A_1), \dots, \nu(A_K))$ follows a Dirichlet distribution with parameters $(\alpha G_0(A_1), \dots, \alpha G_0(A_K))$ ³. It is well known [14] that a sample ν from $DP(\alpha, G_0)$ has the form $\nu = \sum_{c=1}^{\infty} \beta^c \delta_{\bar{\theta}^c}$, where $\delta_{\bar{\theta}}$ is the Dirac measure at point $\bar{\theta} \in \Theta$, the $\bar{\theta}^c$'s are i.i.d. with distribution G_0 and represent the centres of the clusters (indexed by c), and the weights β^c 's are generated using a Beta distribution according to the following stick-breaking construction:

$$\begin{aligned} \tilde{\beta}^c &\sim \text{Beta}(1, \alpha) \text{ (the } \tilde{\beta}^c \text{'s are independent),} \\ \beta^c &= \tilde{\beta}^c \prod_{i=1}^{c-1} (1 - \tilde{\beta}^i). \end{aligned}$$

When $(\theta^u)_{u \in \mathcal{U}}$ is generated under the above DP mixture model, we can compute the distribution of θ^u given $\theta^{\mathcal{U} \setminus u} = (\theta^v)_{v \in \mathcal{U} \setminus \{u\}}$. When $\theta^{\mathcal{U} \setminus u}$ is fixed, then users in $\mathcal{U} \setminus \{u\}$ are clustered and the set of corresponding clusters is denoted by $\mathcal{C}^{\mathcal{U} \setminus \{u\}}$. Users in cluster $c \in \mathcal{C}^{\mathcal{U} \setminus \{u\}}$ share the same transition kernel $\bar{\theta}^c$, and the number of users assigned to cluster c is denoted by $n_{c,-u} = \sum_{u \in \mathcal{U} \setminus \{u\}} \mathbb{1}_{u \in c}$. The distribution of θ^u given $\theta^{\mathcal{U} \setminus u}$ is then:

$$\theta^u | \theta^{\mathcal{U} \setminus u} \sim \begin{cases} G_0 & \text{w.p. } \frac{\alpha}{\alpha + |\mathcal{U}| - 1}, \\ \delta_{\bar{\theta}^c} & \text{w.p. } \frac{n_{c,-u}}{\alpha + |\mathcal{U}| - 1}, \forall c \in \mathcal{C}^{\mathcal{U} \setminus \{u\}}. \end{cases} \quad (3)$$

(3) makes the cluster structure of the DP mixture model explicit. Indeed, when considering a new user u , a new cluster containing user u only is created with probability $\frac{\alpha}{\alpha + |\mathcal{U}| - 1}$, and

³The Dirichlet distribution with parameters $(\alpha_1, \dots, \alpha_K)$ has density (with respect to Lebesgue measure) proportional to $\mathbb{1}(x_1 > 0, \dots, x_K > 0) \mathbb{1}(x_1 + \dots + x_K = 1) \prod_{k=1}^K x_k^{\alpha_k}$.

user u is associated with an existing cluster c with probability proportional to the number of users already assigned to this cluster. Refer to [15] for a more detailed description on DP mixture models.

Our prediction method simply consists in approximating $\mathbb{E}[\theta^u | x^{\mathcal{U}}]$ by the expectation w.r.t. the posterior distribution $g | x^{\mathcal{U}}$. In other words, for user u , the estimated next position will be:

$$\hat{x}^u \in \arg \max_{j \in \mathcal{L}} E_g[\theta_{x_{n_u}, j}^u | x^{\mathcal{U}}], \quad (4)$$

where $E_g[\cdot]$ denotes the expectation w.r.t. the probability measure induced by g . To compute $E_g[\theta^u | x^{\mathcal{U}}]$, we rely on Gibbs sampling techniques to generate samples with distribution $g | x^{\mathcal{U}}$. The way $g | x^{\mathcal{U}}$ concentrates its mass around the true prior μ will depend on the choice of parameters α and G_0 , and to improve the accuracy of our predictor, these parameters will be constantly updated when successive samples are produced.

B. CAMP: Cluster-Aided Mobility Predictor

Next we present CAMP, our mobility prediction algorithm. The objective of this algorithm is to estimate $E_g[\theta^u | x^{\mathcal{U}}]$ from which we derive the predictions according to (4). CAMP consists in generating independent samples of the assignment of users to clusters induced by the posterior distribution $g | x^{\mathcal{U}}$, and then in providing an estimate of $E_g[\theta^u | x^{\mathcal{U}}]$ from these samples. As mentioned above, the accuracy of this estimate strongly depends on the choice of parameters α and G_0 in the DP mixture model, and these parameters will be updated as new samples are generated.

More precisely, the CAMP algorithm consists in two steps. (i) In the first step, we use Gibbs sampler to generate B samples of the assignment of users to clusters under the probability measure induced by $g | x^{\mathcal{U}}$, and update the parameters α and G_0 of the DP mixture model using these samples (hence we update the prior distribution g). We repeat this procedure $K - 1$ times. In the k -th iteration, we construct B samples of users' assignment. The b -th assignment sample is referred to as $c^{\mathcal{U}, b, k} = (c^{u, b, k})_{u \in \mathcal{U}}$ in CAMP pseudo-code, where $c^{u, b, k}$ is the cluster of user u in that sample. The subroutines providing the assignment samples, and updating the parameters of the prior distribution g are described in details in §IV-B1 and §IV-B2, respectively. At the end of the first step, we have constructed a prior distribution g parametrized by G_0^K and α_K which is adapted to the data, i.e., a distribution that concentrates its mass on the true prior μ . (ii) In the second step, we use the updated prior g to generate one last time B samples of users' assignment. Using these samples, we compute an estimate $\hat{\theta}^u$ of $E_g[\theta^u | x^{\mathcal{U}}]$ for each user u , and finally derive the prediction \hat{x}^u of the next position of user u . The way we compute $\hat{\theta}^u$ is detailed in §IV-B3.

The CAMP algorithm takes as inputs the data $x^{\mathcal{U}}$, the number K of updates of the prior distribution g , the number of samples B generated by the Gibbs sampler in each iteration, and the number of times M the users' assignment is updated when producing a single assignment sample using Gibbs

sampler (under Gibbs sampler, the assignment is a Markov chain, which we simulate long enough so as it has the desired distribution). K , B , and M have to be chosen as large as possible. Of course, increasing these parameters also increases the complexity of the algorithm, and we may wish to select the parameters so as to achieve an appropriate trade-off between accuracy and complexity.

Algorithm 1 CAMP

Input: $x^{\mathcal{U}}, K, B, M$

Step 1: Updates of G_0 and α

$G_0^1 \leftarrow \text{Uniform}(\Theta), \alpha_1 \leftarrow 1$

for $k = 1 \dots K - 1$ **do**

for $b = 1 \dots B$ **do**

$c^{\mathcal{U}, b, k} \leftarrow \text{GibbsSampler}(x^{\mathcal{U}}, G_0^k, \alpha_k, M)$

end

$G_0^{k+1}, \alpha_{k+1} \leftarrow \text{UpdateDP}(x^{\mathcal{U}}, G_0^k, \{c^{\mathcal{U}, b, k}\}_{b=1 \dots B})$

end

Step 2: Last sampling and prediction

for $b = 1 \dots B$ **do**

$c^{\mathcal{U}, b, K} \leftarrow \text{GibbsSampler}(x^{\mathcal{U}}, G_0^K, \alpha_K, M)$

end

Compute $\hat{\theta}^u$ by implementing (8) using $\{c^{u, b, K}\}_{b=1, \dots, B}$ and G_0^K

$\hat{x}^u = \arg \max_j \hat{\theta}_{x_{n_u}, j}^u$

Output: $\hat{\theta}^u, \hat{x}^u$

1) *Sampling from the DP mixture posterior:* We use Gibbs sampler [16] to generate independent samples of the assignment of users to clusters under the probability measure induced by the posterior $g | x^{\mathcal{U}}$, i.e., samples of assignment with distribution $P_g[c^{\mathcal{U}} | x^{\mathcal{U}}]$, where P_g denotes the probability measure induced by g . Gibbs sampling is a classical MCMC method to generate samples from a given distribution. It consists in constructing and simulating a Markov chain whose stationary state has the desired distribution. In our case, the state of the Markov chain is the assignment $c^{\mathcal{U}}$, and its stationary distribution is $P_g[c^{\mathcal{U}} | x^{\mathcal{U}}]$. The Markov chain should be simulated long enough (here the number of steps is denoted by M) so that at the end of the simulation, the state of the Markov chain has converged to the steady-state. The pseudo-code of the proposed Gibbs sampler is provided in Algorithm 2, and easily follows from the description of the DP mixture model provided in (3).

To produce a sample of the assignment of users to clusters, we proceed as follows. Initially, we group all users in the same cluster c_1 , the number of cluster N is set to 1, and the number of users (except for user u) $n_{c_1, -u}$ assigned to cluster c_1 is $|\mathcal{U}| - 1$. (see Algorithm 2). Then the assignment is revised M times. In each iteration, each user is considered and assigned to either an existing cluster, or to a newly created cluster (the latter is denoted by c_{N+1} if in the previous iteration there was N clusters). This assignment is made randomly according to the model described in (3). Note that in the definition of β_c , we have $G_0(d\theta | x^c) = \frac{P_\theta(x^c)G_0(d\theta)}{\int_\theta P_\theta(x^c)G_0(d\theta)}$, where x^c corresponds to the data of users in cluster c , i.e., $x^c = (x^u)_{u \in c}$.

Algorithm 2 GibbsSampler

Input: $x^{\mathcal{U}}, G_0, \alpha, M$
 $\forall u \in \mathcal{U}, c^u \leftarrow c_1, n_{c_1, -u} \leftarrow |\mathcal{U}| - 1; N \leftarrow 1; \mathcal{C}^{\mathcal{U}} = \{c_1\}.$
for $i = 1 \dots M$ **do**

 for each $u \in \mathcal{U}$ **do**
 $c^u \leftarrow c^u \setminus \{u\}$
 $\beta_{new} \leftarrow z \frac{\alpha}{\alpha + |\mathcal{U}| - 1} \int_{\theta} P_{\theta}(x^u) G_0(d\theta)$
 $\beta_c \leftarrow z \frac{n_{c, -u}}{\alpha + |\mathcal{U}| - 1} \int_{\theta} P_{\theta}(x^u) G_0(d\theta | x^c), \forall c \in \mathcal{C}^{\mathcal{U}} \setminus \{u\}$

 In the above expressions, z is a normalizing constant, i.e., selected so as $\beta_{new} + \sum_{c \in \mathcal{C}^{\mathcal{U}} \setminus \{u\}} \beta_c = 1$;

 With probability β_{new} do:

 $c_{N+1} \leftarrow \{u\}; c^u \leftarrow c_{N+1}; n_{c_{N+1}, -u} \leftarrow 0;$
 $n_{c_{N+1}, -v} \leftarrow 1, \forall v \neq u; \mathcal{C}^{\mathcal{U}} \leftarrow \mathcal{C}^{\mathcal{U}} \cup \{c_{N+1}\};$
 $N \leftarrow N + 1;$

 and with probability β_c do:

 $c^u \leftarrow c; c \leftarrow c \cup \{u\}; n_{c, -v} \leftarrow n_{c, -v} + 1, \forall v \neq u.$
end
end
Output: $\mathcal{C}^{\mathcal{U}}$

2) *Updates of G_0 and α :* As in any Bayesian inference method, our prediction method could suffer from a bad choice of parameters α and G_0 defining the prior g . For example, by choosing a small value for α , we tend to get a very small number of clusters, and possibly only one cluster. On the contrary, selecting a too large α would result in a too large number of clusters, and in turn, would make our algorithm unable to capture similarities in the mobility patterns of the various users. To circumvent this issue, we update and fit the parameters to the data, as suggested in [8]. In the CAMP algorithm, the initial base distribution is uniform over all transition kernels (over Θ) and α is taken equal to 1. Then after each iteration, we exploit the samples of assignments of users to clusters to update these initial parameters, by refining our estimates of G_0 and α .

Algorithm 3 UpdateDP at the k -th iteration

Input: $x^{\mathcal{U}}, G_0^k, \{\mathcal{C}^{\mathcal{U}, b, k}\}_{b=1, \dots, B}$

 Compute $G_0^{k+1}(\cdot)$ and α_{k+1} as follows.

$$G_0^{k+1}(\cdot) = \frac{1}{B} \sum_{b=1}^B \sum_{c \in \mathcal{C}^{\mathcal{U}, b, k}} \frac{n_{c, b, k}}{|\mathcal{U}|} G_0^k(\cdot | x^c) \quad (5)$$

$$\alpha_{k+1} = \arg \min_{\alpha \in \mathbb{R}} \left| \sum_{i=1}^{|\mathcal{U}|} \frac{\alpha}{\alpha + i - 1} - \frac{1}{B} \sum_{b=1}^B N_b \right| \quad (6)$$

where $n_{c, b, k}$ is the size of cluster $c \in \mathcal{C}^{\mathcal{U}, b, k}$, and N_b is the total number of (non-empty) clusters in $\mathcal{C}^{\mathcal{U}, b, k}$.

Output: G_0^{k+1}, α_{k+1}

Note that (5) simply corresponds to a kernel density estimator based on the B cluster samples obtained with prior distribution parametrized by G_0^k and α_k , whereas (6) corresponds to a maximum likelihood estimate (see [17]), which sets α_{k+1} to the value which is most likely to have resulted in the average

number of clusters obtained when sampling from the model with parameters G_0^k and α_k .

3) *Computation of $\hat{\theta}^u$:* As mentioned earlier, $\hat{\theta}^u$ is an estimator of $E_g[\theta^u | x^{\mathcal{U}}]$, where g is parameterized by G_0^K and α_K , and is used for our prediction of user- u 's mobility. $\hat{\theta}^u$ is just the empirical average of $\bar{\theta}^c$ for clusters c to which user- u is associated in the B last samples generated in CAMP, i.e.,

$$\hat{\theta}^u = \frac{1}{B} \sum_{b=1}^B E_g[\bar{\theta}^{c^{u, b, K}} | x^{c^{u, b, K}}] \quad (7)$$

$$= \frac{1}{B} \sum_{b=1}^B \frac{\int_{\theta} \theta \cdot P_{\theta}(x^{c^{u, b, K}}) G_0^K(d\theta)}{\int_{\theta} P_{\theta}(x^{c^{u, b, K}}) G_0^K(d\theta)}. \quad (8)$$

Note that in view of the law of large numbers, when B grows large, $\hat{\theta}^u$ converges to $E_g[\theta^u | x^{\mathcal{U}}]$. The predictions for user u are made by first computing an estimated transition kernel $\hat{\theta}^u$ according to (8). We derive an explicit expression of $\hat{\theta}^u$ that does not depend on G_0^K , but only on data and the samples generated in the CAMP algorithms. This expression, given in the following lemma, will be useful to understand to what extent the prediction of user- u 's mobility under CAMP leverages observed trajectories of other users.

Lemma 1 For any i, j , $\hat{\theta}_{i, j}^u$ is computed by a weighted sum of all users' empirical transition kernels $(n_{i, j}^v / n_i^v, v \in \mathcal{U})$, i.e.,

$$\hat{\theta}_{i, j}^u = \eta_i + \sum_{v \in \mathcal{U}} \gamma_i^v \frac{n_{i, j}^v}{n_i^v}, \quad (9)$$

$$\text{where } \eta_i = \sum_{\substack{c_1 \dots c_K: \\ u \in c_K}} \xi_{c_1 \dots c_K} \frac{1}{|\mathcal{L}| + \sum_{k=1}^K n_i^{c_k}} \frac{|\mathcal{U}|}{n_{c_K}^K} \prod_{k=1}^K \omega_{c_k}^k,$$

$$\gamma_i^v = \sum_{\substack{c_1 \dots c_K: \\ u \in c_K}} \xi_{c_1 \dots c_K} \frac{n_i^v \sum_{k=1}^K \mathbb{1}(v \in c_k)}{|\mathcal{L}| + \sum_{k=1}^K n_i^{c_k}} \frac{|\mathcal{U}|}{n_{c_K}^K} \prod_{k=1}^K \omega_{c_k}^k.$$

The sum $\sum_{c_1 \dots c_K}$ stands for $\sum_{c_1 \in \mathcal{C}_1} \dots \sum_{c_K \in \mathcal{C}_K}$, and \mathcal{C}_k is the set of every cluster sampled at k -th iterations (i.e., $\mathcal{C}_k = \{c | \sum_{b=1}^B \sum_{u \in \mathcal{U}} \mathbb{1}(c^{u, b, k} = c) > 0\}$). ω_c^k and $\xi_{c_1 \dots c_K}$ are given by:

$$\xi_{c_1 \dots c_K} = \prod_{i \in \mathcal{L}} \frac{\prod_{j \in \mathcal{L}} \Gamma(1 + \sum_{k=1 \dots K} n_{i, j}^{c_k})}{\Gamma(|\mathcal{L}| + \sum_{k=1 \dots K} n_i^{c_k})},$$

$$\omega_c^K = \frac{n_c^K}{B |\mathcal{U}| \sum_{c_1 \dots c_{K-1}} \xi_{c_1 \dots c_{K-1}, c} \prod_{k=1}^{K-1} \omega_{c_k}^k},$$

where $n_{i, j}^c = \sum_{u \in c} n_{i, j}^u$, $n_i^c = \sum_{j \in \mathcal{L}} n_{i, j}^c$, and $n_c^k = \sum_{b=1}^B \sum_{u \in \mathcal{U}} \mathbb{1}(c^{u, b, k} = c)$.

Proof. Refer to Appendix. \square

When the current location i is fixed, the first term in the r.h.s. of (9) is constant over all users. The second term can be interpreted as a weighted sum of the empirical transition kernels of all users (i.e., $n_{i, j}^v / n_i^v, \forall v \in \mathcal{U}$). The weight of user v (γ_i^v in (9)) quantifies how much we account for user- v 's trajectory in the prediction for user u at the current location i , and can be seen as a notion of similarity between v and

u . Indeed, as the number of sampled clusters in which both u and v are involved increases, γ_i^v in (9) increases accordingly. Also, if v has relatively high n_i^v compared to other users (i.e., v has accumulated more observations at the location i than other users), a higher weight is assigned to v .

4) *Estimating the Staying-times*: Next we provide a way of estimating how long user u will stay at her current location i . We may perform such estimation when the available data include the time users stay at the various locations. Typically, the existing spatio-temporal predictors predict the staying time at the current location $x_{n_u}^u$ by computing average [5] or p -quantile [3] of user u 's staying times observed at her previous visits to $x_{n_u}^u$. On the other hand, CAMP additionally exploits other users' staying time observations using the weight γ_i^v . More precisely, the staying time of user u at location $x_{n_u}^u$ (denoted by $\hat{s}_{n_u}^u$) is estimated by

$$\hat{s}_{n_u}^u = \sum_{v \in \mathcal{U}} z \gamma_i^v \frac{1}{n_i^v} \sum_{t: x_t^v = i} s_t^v, \text{ where } i = x_{n_u}^u. \quad (10)$$

z in (10) is a normalization constant to make the sum of weights over all users equal to 1. The estimate (10) is a heuristic, for γ_i^v is actually obtained by clustering based on their location trajectories $x^{\mathcal{U}}$, rather than their staying times. This heuristic estimate actually performs well as empirically shown in Section VI-B4.

V. CONSISTENCY OF CAMP PREDICTOR

In this section, we analyze to what extent $E_g[\theta^u | x^{\mathcal{U}}]$ (that is well approximated, when B is large, by $\hat{\theta}^u$ derived in the CAMP algorithm) is close to $\mathbb{E}[\theta^u | x^u]$, the expectation under the true prior μ . We are mainly interested in the regime where the user population \mathcal{U} becomes large, while the number of observations n^u for each user remains bounded. This regime is motivated by the fact it is often impractical to gather long trajectories for a given user, while the user population available may on the contrary be very large. For the sake of the analysis, we assume that the length n^u of user- u 's observed trajectory is a random variable with distribution $p \in \mathcal{P}(\mathbb{N})$, and that the lengths of trajectories are independent across users. We further assume that the length is upper bounded by \bar{n} , e.g., $\bar{n} = \max\{n : p(n) > 0\} < \infty$.

Since the length of trajectories is bounded, we cannot ensure that $|E_g[\theta^u | x^{\mathcal{U}}] - \mathbb{E}[\theta^u | x^u]|$ is arbitrarily small. Indeed, for example if users' trajectories are of length 2 only, we cannot group users into clusters, and in turn, we can only get a precise estimate of the transition kernels averaged over all users. In particular, we cannot hope to estimate $\mathbb{E}[\theta^u | x^{\mathcal{U}}]$ for each user u . Next we formalize this observation. We denote by $\mathcal{H}_{\bar{n}} \subseteq \mathcal{L}^{\bar{n}}$ the set of possible trajectories of length less than \bar{n} . With finite-length observed trajectories, there are distributions $\nu \in \mathcal{P}(\Theta)$ that cannot be distinguished from the true prior μ by just observing users' trajectories, i.e., these distributions induce the same law on the observed trajectories as μ : $P_\nu = \mathbb{P}$ on $\mathcal{H}_{\bar{n}}$ (here P_ν denotes the probability measure induced under ν , and recall that \mathbb{P} is the probability measure induced by μ). We

prove that, when the number of observed users grows large, $|E_g[\theta^u | x^{\mathcal{U}}] - \mathbb{E}[\theta^u | x^u]|$ is upper-bounded by the performance provided by a distribution ν indistinguishable from μ , which expresses the consistency of our inference framework. Before we state our result, we introduce the following two notions:

KL ϵ -neighborhood: the Kullback-Leibler ϵ -neighborhood $K_{\epsilon, \bar{n}}(\mu)$ of a distribution $\mu \in \mathcal{P}(\Theta)$ with respect to $\mathcal{H}_{\bar{n}}$ is defined as the following set of distributions:

$$K_{\epsilon, \bar{n}}(\mu) = \{\nu \in \mathcal{P}(\Theta) : KL_{\bar{n}}(\mu, \nu) < \epsilon\},$$

where $KL_{\bar{n}}(\mu, \nu) = \sum_{x \in \mathcal{H}_{\bar{n}}} P_\mu(x) \log \frac{P_\mu(x)}{P_\nu(x)}$.

KL support: The distribution μ is in the Kullback-Leibler support of a distribution $g \in \mathcal{P}(\mathcal{P}(\Theta))$ with respect to $\mathcal{H}_{\bar{n}}$ if $g(K_{\epsilon, \bar{n}}(\mu)) > 0$ for all $\epsilon > 0$.

Theorem 2 If $\mu \in \mathcal{P}(\Theta)$ is in the KL-support of g with respect to $\mathcal{H}_{\bar{n}}$, then we have, μ -almost surely, for any $i, j \in \mathcal{L}$,

$$\begin{aligned} & \lim_{|\mathcal{U}| \rightarrow \infty} |E_g[\theta_{i,j}^u | X^{\mathcal{U}}] - \mathbb{E}[\theta_{i,j}^u | X^u]| \\ & \leq \sup_{\substack{\nu \in \mathcal{P}(\Theta) \\ P_\nu = \mathbb{P} \text{ on } \mathcal{H}_{\bar{n}}}} |E_\nu[\theta_{i,j}^u | X^u] - \mathbb{E}[\theta_{i,j}^u | X^u]|. \end{aligned} \quad (11)$$

Proof. Refer to Appendix. \square

The r.h.s. of (2) captures the performance of an algorithm that would perfectly estimate $E_\nu[\theta^u | X^u]$ for the worst distribution ν , which agrees with the true prior μ on $\mathcal{H}_{\bar{n}}$. Note that in our framework, for the prior $g \in \mathcal{P}(\mathcal{P}(\Theta))$, we use is a DP mixture $DP(G_0, \alpha)$, with a base measure $G_0 \in \mathcal{P}(\Theta)$ having full support Θ . Therefore, the KL-support of g is here the whole space $\mathcal{P}(\Theta)$; it thus contains μ .

As far as we are aware, Theorem 2 presents the first performance result on inference algorithms using DP mixture models with *indirect* observations. By indirect observations, we mean that the kernels $(\theta^u)_{u \in \mathcal{U}}$ cannot be observed directly, but are revealed only through the trajectories $x^{\mathcal{U}}$. Most existing analysis [18]–[20] do not apply in our setting, as these papers aim at identifying conditions on the Bayesian prior g and on the true distribution μ under which the Bayesian posterior $g|\theta^{\mathcal{U}}$ will converge (either weakly or in L_1 -norm) to μ in the limit of large population size. Hence, existing analysis are concerned with *direct* observations of the kernels $(\theta^u)_{u \in \mathcal{U}}$.

VI. EMPIRICAL EVALUATION OF CAMP

A. Mobility Traces

We evaluate the performance of CAMP predictor using two sets of mobility traces collected on a Wi-Fi and cellular network, respectively.

Wi-Fi traces [21]. We use the dataset of [21] where the mobility of 62 users are collected for three months in Wi-Fi networks mainly around a campus in South Korea. The smartphone of each users periodically scans its radio environment and gets a list of mac addresses of available access points (APs). To map these lists of APs collected over time to a set of locations, we compute the Jaccard index⁴ between two lists

⁴Jaccard index between two lists A and B is defined as $\frac{|A \cap B|}{|A \cup B|}$.

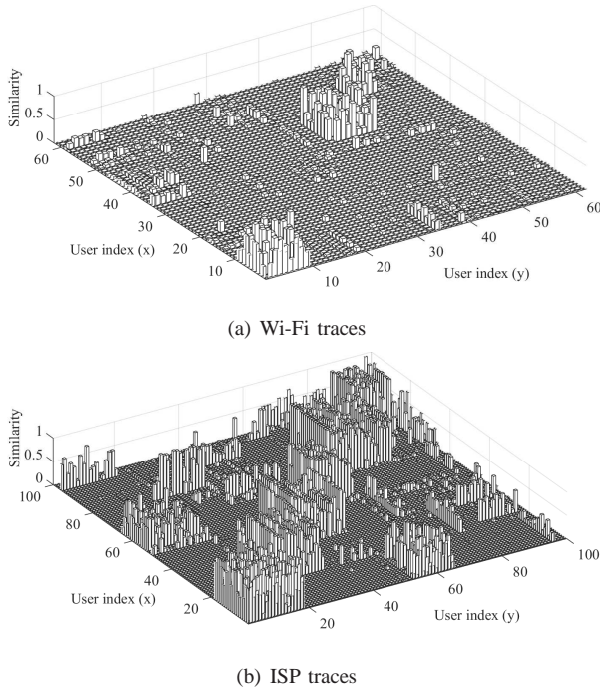


Fig. 1. Similarities between pairs of users. For the ISP traces we restrict the plot to 100 randomly selected users.

of of APs scanned at different times. If two lists of APs have a Jaccard index higher than 0.5, these two lists are considered to correspond to a same geographical locations [21]. From the constructed set of locations, we then construct the trajectories of the various users.

ISP traces [22]. We also use the call detailed record (CDR) dataset provided by Orange where the mobility of 50000 subscribers in Senegal are measured over two weeks. We use the SET2 data [22], where the mobility of a given user is reported as a sequence of base station (BS) ids, and time stamps. Each record is obtained only when the user communicates with base stations (e.g., phone call, text message).

In each dataset, we first restrict our attention to a subset \mathcal{L} of frequently visited locations. We select the 116 and 80 most visited locations in Wi-Fi traces and ISP traces datasets, respectively. We then re-construct users' trajectories by removing locations not in \mathcal{L} . For the ISP dataset, we extract 200 users (randomly chosen among users who visited at least 10 of the locations in \mathcal{L}). From the re-constructed trajectories, we observe a total number of transitions from one location to another equal to 8194 and 13453 for the Wi-Fi and ISP dataset.

Users' similarity. Before actually evaluating the performance of various prediction algorithms, we wished to assess whether users exhibit similar mobility patterns, that could in turn be exploited in our predictions. Here, we test the similarity of pairs of users only. More precisely, we wish to know whether the observed trajectory of user v could be aggregated to that of user u to improve the prediction of user- u 's mobility. To this aim, we use the concept of mutual prediction [23] as follows.

We first define the empirical accuracy of an estimator $\hat{\theta}$ of

user- u 's transition kernel:

$$\overline{AC}^u(\hat{\theta}) = \frac{1}{n^u - 1} \sum_{t=2}^{n^u} \mathbb{1}(x_t^u = \arg \max_j \hat{\theta}_{x_{t-1}^u, j}) \quad (12)$$

Let $\hat{\theta}^{u*}$ be the maximum likelihood estimator of θ^u given x^u (i.e., $\hat{\theta}_{i,j}^{u*} = \frac{n_{i,j}^u}{n_i^u}, \forall i, j \in \mathcal{L}$). Intuitively, user- v 's trajectory is useful to predict the mobility of user u if $\hat{\theta}^{v*}$ has a high empirical accuracy for user u , i.e., if $\overline{AC}^u(\hat{\theta}^{v*})$ is high. We hence define the *similarity* $\text{sim}(u, v)$ of users u and v as $\text{sim}(u, v) = \overline{AC}^u(\hat{\theta}^{v*}) / \overline{AC}^u(\hat{\theta}^{u*})$. Note that the notion of similarity is not symmetric (in general $\text{sim}(u, v) \neq \text{sim}(v, u)$), and it always takes its value between 0 and 1.

Fig. 1 (a) and (b) present the similarity between 62 users in Wi-Fi trace and 100 users in the ISP subscriber dataset. To provide meaningful plots, we have ordered users so that pairs of users with high similarity are neighbours (to this aim, we have run the spectral clustering algorithm [11] and re-grouped users in the identified clusters). From these plots, the similarity of users is apparent, however we also clearly observe that perfect clusters (in which users' patterns are exactly same) do not really exist. From the dataset, we observe that 1.65% and 5% of user pairs out of all possible pairs have similarity higher than 0.5 for the Wi-Fi and ISP traces. We also computed the number of users having at least one user with whom the similarity is higher than 0.5. In the Wi-Fi traces, we found 19 (out of 62) such users, whereas in the ISP traces there are 173 (out of 200) such users. These numbers are high, and justify the design of cluster-aided predictors.

B. Prediction Accuracy

1) Tested Predictors: We assess the performance of six types of predictors: the order-1 Markov predictor (Markov [4]), the order-2 Markov predictor with fallback (Markov-O(2) [4]), AGG, CAMP and CAMP^C , AGG^C . Before describing each predictor, we briefly introduce some notations regarding the training data available at a given time. The time stamp of the arrival at t -th location on user- u 's trajectory is denoted by $d_t^u \in \mathbb{R}$, and $n^u(d)$ is the length of user- u 's trajectory collected before time d (i.e., $n^u(d) = \max\{s | d_s^u < d\}$). The collection of users' trajectories available for a prediction at time d is denoted by $x^{\mathcal{U}, d}$ (i.e., $x^{\mathcal{U}, d} = (x^{v, d})_{v \in \mathcal{U}}$, where $x^{v, d} = (x_1^v, \dots, x_{n^v(d)}^v)$). The prediction for x_t^u is denoted by \hat{x}_t^u .

In order to derive an estimate of the t -th location \hat{x}_t^u of user u , the Markov predictors first estimate θ^u based on user- u trajectory only, i.e., based on x^{u, d_t^u} . In contrast, AGG and CAMP algorithms exploit the data available on all users $x^{\mathcal{U}, d_t^u}$ to estimate θ^u . The AGG algorithm tries in a very naive way to exploit users' similarities. It considers that all users have the same transition kernel (as if there were a single cluster only), and thus uses all trajectories (in the same way) to estimate θ_u . CAMP^C (resp. AGG^C) differs from CAMP (resp. AGG) in that its prediction at time d under for user u uses other users' complete trajectories (i.e., $x^{\mathcal{U} \setminus u}$). This corresponds to a case where user u starts moving along her trajectory after other users have gathered sufficiently long trajectories.

TABLE I
ORDER-1 PREDICTORS.

	$\hat{\theta}^{u,d_t^u}$	\hat{x}_t^u
Markov [4]	$\arg \max_{\theta} P_{\theta}(x^{u,d_t^u})$	$\arg \max_j \hat{\theta}_{x_{t-1}^u, j}^{u,d_t^u}$
AGG	$\arg \max_{\theta} P_{\theta}(x^{u,d_t^u})$	
CAMP	$\simeq E_g[\theta^u x^{u,d_t^u}]$	
AGG ^C	$\arg \max_{\theta} P_{\theta}(x^{u \setminus u}, x^{u,d_t^u})$	
CAMP ^C	$\simeq E_g[\theta^u x^{u \setminus u}, x^{u,d_t^u}]$	

Under all algorithms, the estimated θ^u is denoted by $\hat{\theta}^{u,d_t^u}$. Finally, Markov-O(2) assumes that users' trajectories are order-2 Markov chains, and for the locations where the corresponding order-2 transitions are not observed, Markov-O(2) falls back to the Markov predictor. The description of the various predictors is summarized in Table 1.

The parameters B , K and M for CAMP and CAMP^C are set to 8, 3 and 30.

2) *Results*: We assess the performance of the various algorithms using two main types of metrics. The first metric, referred to as the Cumulative Accurate Prediction Ratio (CAPR), is defined as the fraction of accurate predictions for all users up to time d :

$$CAPR_{time} = \frac{1}{\sum_{u \in \mathcal{U}} (n^u(d) - 1)} \sum_{u \in \mathcal{U}} \sum_{s=2}^{n^u(d)} \mathbb{1}(\hat{x}_s^u = x_s^u).$$

We also introduce a similar metric that captures the cumulative accuracy of predictions after observing t different locations on users' trajectories:

$$CAPR = \frac{1}{(t-1) \sum_{u \in \mathcal{U}} \mathbb{1}(n^u \geq t)} \sum_{\substack{u \in \mathcal{U} \\ n^u \geq t}} \sum_{s=2}^t \mathbb{1}(\hat{x}_s^u = x_s^u).$$

The second type of metrics concerns the instantaneous accuracy of the predictions. The Instantaneous Accurate Prediction Ratio (IAPR) after observing t different locations on users' trajectories is defined as follows.

$$IAPR = \frac{1}{\sum_{u \in \mathcal{U}} \mathbb{1}(n^u \geq t)} \sum_{u \in \mathcal{U}, n^u \geq t} \frac{n_{x_{t-1}^u, \hat{x}_t^u}^u}{n_{x_{t-1}^u}^u}.$$

Fig.2(a)-(b) present $CAPR_{time}$ as a function of time d for various algorithms and for the two mobility traces. CAMP outperforms all other algorithms at any time. The improvement over Markov and Markov-O(2) can be as high as 65%. This illustrates the performance gain that can be achieved when exploiting users' similarities. Note Markov-O(2) does not outperform Markov, which was also observed in [3]. In the following, we only evaluate the performance of the Markov predictor, and do not report that of its order-2 equivalent.

In Fig.2 (c)-(f), we plot the CAPR and IAPR as a function of the length t of the observed trajectory. In Fig.2(c) and (d), when the collected trajectory is not sufficient (i.e., $t = 10$), CAMP^C and CAMP outperforms Markov by 64% and 40%, respectively. Regarding the IAPR in Wi-Fi traces, Fig 2(e) shows that CAMP and CAMP^C provide much better predictions than Markov,

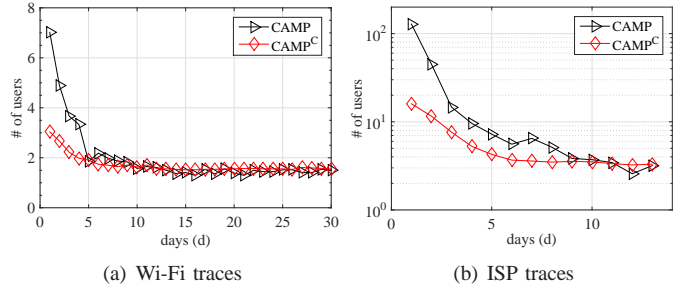


Fig. 3. Number of u -similar users, averaged over all users u , vs. time.

when the length of trajectory is less than 140. After a sufficient training data is collected, they yield comparable IAPR. In Fig 2 (f), for the ISP traces, the IAPR under CAMP and Markov are similar sooner, for trajectories of length greater than 20 only.

In Fig.2 (g) and (h), we evaluate the CAPR and IAPR averaged only over users having at least one user with whom the similarity is higher than 0.5 (see §VI-A). These users are referred to as Mobility Friendly (MF) users. In Fig.2(g), we observe that for MF users, the gain of CAMP^C and CAMP becomes really significant, i.e., when $t=10$, the CAPR of CAMP^C and CAMP outperform that of Markov by 102% and 65%, respectively. Also note that CAMP^C becomes significantly better than CAMP for MF users. This is explained by the fact that we can predict the mobility of MF users much more accurately if we have a long history of the mobility of users they are similar to. The performance for MF users in the ISP traces is not presented, because there, most of users (i.e., 86%) are already MF users.

3) *Exploiting Similarities in CAMP*: Recall that, by the weight of the empirical transition kernel of user v (i.e., γ_i^v) in computing $\hat{\theta}^u$ in (9), we can quantify to what extent the observed trajectory of user v is taken into account in the estimate $\hat{\theta}^u$. When summing γ_i^v over all locations i , we get an aggregate indicator capturing how v impacts the prediction for user- u 's mobility. To understand how many users actually impact the prediction for user u in the CAMP, we may look at the cardinality of the set of users whose aggregate indicator exceeds a given threshold: $\{v | z \sum_{i \in \mathcal{L}} \gamma_i^v > \frac{1}{|\mathcal{U}|}\}$, where z is a normalization constant to make the sum of aggregate indicators over all users equal to 1. The above set is called the set of u -similar users.

In Fig.3, we plot the number of u -similar users, averaged over all users u , and as a function of the length of trajectories (in days d). In case of CAMP, the first day, the average numbers are 7 and 110 in Wi-Fi traces and ISP traces, which means that CAMP aggressively uses the trajectories of all users for its prediction. When the length of the trajectories increase, the average size decreases to 1.5 after one month in Wi-Fi traces and 2.2 after two weeks in ISP traces. In other words, as data is accumulated, CAMP does not use the trajectories of a lot of users for its prediction. This illustrates the adaptive nature of CAMP, which only exploits similarities among users if this is needed. In the case of CAMP^C, we observe a faster decrease

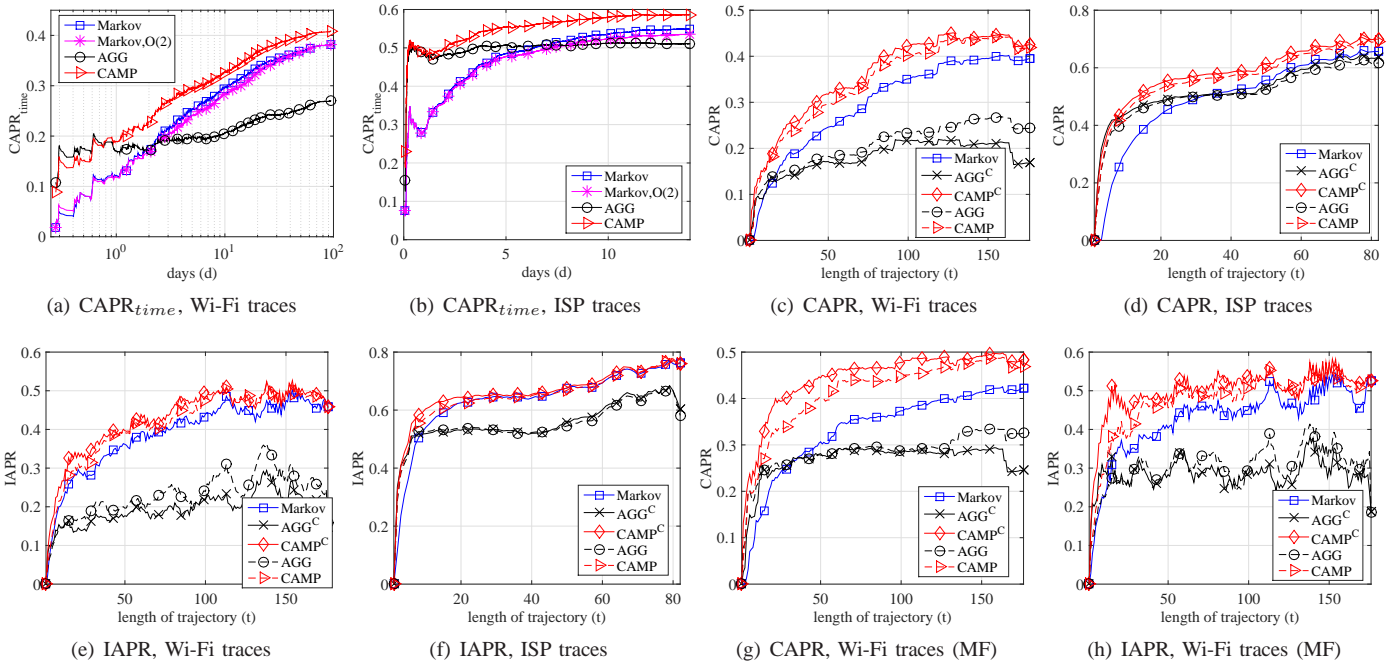


Fig. 2. Performance of various predictors.

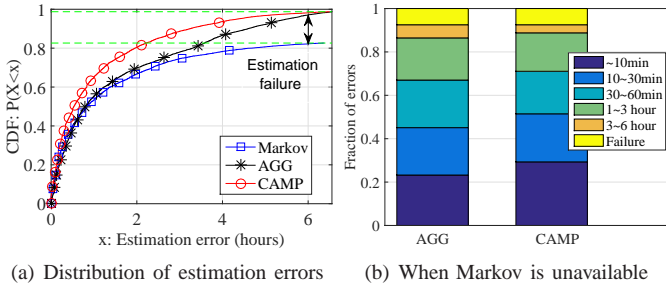


Fig. 4. Estimation error of staying time in Wi-Fi trace. The dashed lines in (a) indicate the fraction of cases where the related training data was collected.

with time of the average number of u -similar users, which means that $CAMP^C$ tends to utilize other users' data more selectively, even at the beginning. This explains why $CAMP^C$ performs better than CAMP in Fig.2.

4) *Error of Staying Time Estimation*: In our scenario, where each user u arrives at t -th location x_t^u , a predictor estimates the staying time \hat{s}_t^u with the available data. Markov predictor [3], [5](resp. AGG) computes the average of staying times of user u (resp. all users) which have been measured at the location x_t^u until d_t^u . CAMP predicts \hat{s}_t^u by computing the equation (10) with the observed data of all users. The performance metric for each user u measured at t -th location is the difference between the estimated and actual staying time (i.e., $|\hat{s}_t^u - s_t^u|$). We call it as *estimation error*. We test the estimation error only with Wi-Fi trace, because we cannot precisely observe staying time in ISP trace in which a location is recorded not periodically, but only when users randomly communicate with base stations.

Fig.4 (a) plots CDFs of estimation errors of every user u and t obtained by tested predictors. CAMP provides lower

estimation error than that of Markov and AGG. The median of CAMP is less than those of Markov and AGG by 35% and 28%, respectively. For 18% of all instances (marked as “Estimation failure”), Markov couldn't provide estimations, because the individual users haven't collected their staying times at the current location before. However in those cases AGG and CAMP are still able to estimate the staying time by using other users' observations. In Fig.4 (b), we further test the estimation quality of AGG and CAMP, when Markov is unavailable due to lack of the individual training data. In that case, 43% of estimations provided by CAMP give less than 30 minutes errors. Median of estimation errors of CAMP is 13.4% less than that of AGG, because CAMP selectively utilizes other users' data.

VII. CONCLUDING REMARKS

In this paper, we have presented a cluster-aided inference method to predict the mobility of users in wireless networks. This method significantly departs from existing prediction techniques, as it aims at exploiting similarities in the mobility patterns of the various users to improve the prediction accuracy. The proposed algorithm, CAMP, relies on Bayesian non-parametric estimation tools, and is robust and adaptive in the sense that it exploits users' mobility similarities only if the latter really exist. We have shown that our Bayesian prediction framework can asymptotically achieve the performance of an optimal predictor when the user population grows large, and have presented extensive experiments indicating that CAMP outperforms any other existing prediction algorithms. Note also that CAMP can be implemented without damaging users' privacy (the data can be anonymized).

Many interesting questions remain about the design of CAMP. In particular, we plan to investigate how to set its parameters (B , K , and M) to achieve an appropriate trade-off between accuracy and complexity. These parameters could also be modified in an online manner while the algorithm is running to adapt to the nature of the data. We further plan to apply the techniques developed in this paper to various kind of *mobility*, e.g., we could investigate how users dynamically browse the web, and use our framework to predict the next visited webpage.

APPENDIX

A. Proof of Lemma 1

Observe that in view of (5), we have:

$$G_0^{k+1}(\mathrm{d}\theta) = \sum_c \omega_c^k P_\theta(x^c) G_0^k(\mathrm{d}\theta), \quad (13)$$

where the sum is over all possible partitions of the set of users \mathcal{U} in clusters and the weight ω_c^k is

$$\omega_c^k = \frac{n_c^k}{B|\mathcal{U}| \int_\theta P_\theta(x^c) G_0^k(\mathrm{d}\theta)}, \quad (14)$$

with $n_c^k = \sum_{b=1}^B \sum_{u \in \mathcal{U}} \mathbb{1}(c^{u,b,k} = c)$. Recursively replacing G_0^k in (13) with G_0^{k-1} and putting $G_0^1 = \text{Uniform}(\Theta)$, we obtain another expression of G_0^K as

$$G_0^K(\mathrm{d}\theta) = \sum_{c_1, \dots, c_{K-1}} \prod_{k=1}^{K-1} \omega_{c_k}^k P_\theta(x^{c_k}) \mathrm{d}\theta, \quad (15)$$

where the sum $\sum_{c_1, \dots, c_{K-1}}$ is $\sum_{c_1 \in \mathcal{C}_1} \dots \sum_{c_{K-1} \in \mathcal{C}_{K-1}}$ where \mathcal{C}_k is a set of every cluster sampled at k -th iterations, i.e., $\mathcal{C}_k = \{c | \sum_{b=1}^B \sum_{u \in \mathcal{U}} \mathbb{1}(c^{u,b,k} = c) > 0\}$. We can further obtain the recursive expression of the weights ω_c^K by plugging (15) in (14):

$$\omega_c^K = \frac{n_c^K}{B|\mathcal{U}| \sum_{c_1, \dots, c_{K-1}} \xi_{c_1, \dots, c_{K-1}, c} \prod_{k=1}^{K-1} \omega_{c_k}^k}, \quad (16)$$

$$\xi_{c_1, \dots, c_K} = \int_\theta \prod_{k=1..K} P_\theta(x^{c_k}) \mathrm{d}\theta \quad (17)$$

$$= \prod_{i \in \mathcal{L}} \frac{\prod_{j \in \mathcal{L}} \Gamma(1 + \sum_{k=1..K} n_{i,j}^{c_k})}{\Gamma(|\mathcal{L}| + \sum_{k=1..K} n_i^{c_k})}, \quad (18)$$

where $n_{i,j}^c = \sum_{u \in c} n_{i,j}^u$, $n_i^c = \sum_{j \in \mathcal{L}} n_{i,j}^c$.

Then, using equations (8), (15) and (17), we get an expression for $\hat{\theta}_{i,j}^u$: In (8), replacing the denominator of each sample b with $\omega_{c^{u,b,K}}^K$, and plugging (15) into numerator, we arrive at

$$\begin{aligned} \hat{\theta}_{i,j}^u &= \frac{1}{B} \sum_{b=1}^B \frac{\omega_{c^{u,b,K}}^K B|\mathcal{U}|}{n_{c^{u,b,K}}^K} \sum_{c_1, \dots, c_{K-1}} \int_\theta \theta_{i,j} P_\theta(x^{c^{u,b,K}}) \\ &\quad \prod_{k=1}^{K-1} P_\theta(x^{c_k}) \omega_{c_k}^k \mathrm{d}\theta \\ &= \sum_{\substack{c_1, \dots, c_K \\ : u \in c_K}} \xi_{c_1, \dots, c_K} \frac{1 + \sum_{k=1}^K n_{i,j}^{c_k}}{|\mathcal{L}| + \sum_{k=1}^K n_i^{c_k}} \frac{|\mathcal{U}|}{n_{c_K}^K} \prod_{k=1..K} \omega_{c_k}^k \end{aligned} \quad (19)$$

Rearranging (19), we arrive at (9).

B. Proof of Theorem 2

The proof of Theorem 2 relies on the following two lemmas.

Lemma 3 If $\mu \in \mathcal{P}(\Theta)$ is in the KL-support of g with respect to $\mathcal{H}_{\bar{\pi}}$, then $g(K_{\epsilon, \bar{\pi}}(\mu) | X^{\mathcal{U}}) \xrightarrow{|\mathcal{U}| \rightarrow \infty} 1$ for all $\epsilon > 0$, μ -almost surely.

The above lemma is a perfect analog of a similar statement for Bayesian consistency with direct observations (see [20], Theorem 6.1 and its corollary). The proof also goes through essentially in the same way; therefore, we do not provide it here. This first lemma states that the set $K_{\epsilon, \bar{\pi}}^C(\mu)$, i.e., the set of distributions ν that do not agree with the true prior μ on $\mathcal{H}_{\bar{\pi}}$ according to the KL distance $KL_{\bar{\pi}}(\mu, \nu)$ w.r.t. $\mathcal{H}_{\bar{\pi}}$, has a vanishing mass under the posterior distribution $g | X^{\mathcal{U}}$, μ -a.s. However, this does not guaranty that the set of distributions ν with $0 < KL_{\bar{\pi}}(\mu, \nu) \leq \epsilon$ will have a negligible impact on the estimates $E_g[\theta^u | X^{\mathcal{U}}]$. Indeed, for this we need continuity with respect to the KL distance over $\mathcal{H}_{\bar{\pi}}$, which the next lemma provides.

Lemma 4 Under the assumptions of Lemma 3, for any bounded continuous $f : \Theta \rightarrow \mathbb{R}$,

$$\lim_{\epsilon \rightarrow 0} \sup_{\nu \in K_{\epsilon, \bar{\pi}}(\mu)} |E_\nu[f] - \mathbb{E}[f]| = \sup_{\substack{\nu \in \mathcal{P}(\Theta) \\ P_\nu = P_\mu \text{ on } \mathcal{H}_{\bar{\pi}}}} |E_\nu[f] - \mathbb{E}[f]|.$$

Proof. Let ρ be the metric on Θ . We use the associated Wasserstein metric d_ρ on $\mathcal{P}(\Theta)$:

$$d_\rho(\mu, \nu) = \inf_{\substack{\pi \in \mathcal{P}(\Theta^2) \\ \pi_1 = \mu, \pi_2 = \nu}} \left\{ \int_{(\theta, \lambda)} \rho(\theta, \lambda) \pi(\mathrm{d}\theta, \mathrm{d}\lambda) \right\},$$

where π_1 and π_2 are the first and second marginals of π , respectively. It is well-known (see [24]) that the space $(\mathcal{P}(\Theta), d_\rho)$ is compact, complete and separable, as (Θ, ρ) is.

Let $\delta > 0$ and let $(\epsilon_k) \in \mathbb{R}_+^{\mathbb{N}}$ be a sequence converging to 0. For all $k \in \mathbb{N}$, let $\nu_k \in \mathcal{P}(\Theta)$ such that

$$|E_{\nu_k}[f] - \mathbb{E}[f]| \geq \sup_{\substack{\nu \in \mathcal{P}(\Theta) \\ KL_{\bar{\pi}}(\mu, \nu) \leq \epsilon_k}} |E_\nu[f] - \mathbb{E}[f]| - \delta.$$

By compactness of $(\mathcal{P}(\Theta), d_\rho)$, there exists a converging subsequence $(\tilde{\nu}_k)$ of (ν_k) , and a corresponding subsequence $(\tilde{\epsilon}_k)$ of (ϵ_k) ; let us call $\nu_\infty \in \mathcal{P}(\Theta)$ its limit. Clearly, we have $D_{n^+}(\mu, \nu_\infty) = 0$. Because the Wasserstein distance metrizes weak convergence (see Theorem 6.9 in [24]) and f is bounded and continuous, we have that $\lim_{k \rightarrow \infty} E_{\tilde{\nu}_k}[f] = E_{\nu_\infty}[f]$. Thus,

$$\begin{aligned} \sup_{\substack{\nu \in \mathcal{P}(\Theta) \\ P_\nu = P_\mu \text{ on } \mathcal{H}_{\bar{\pi}}}} |E_\nu[f] - \mathbb{E}[f]| &\geq |E_{\nu_\infty}[f] - \mathbb{E}[f]| \\ &= \lim_{k \rightarrow \infty} |E_{\tilde{\nu}_k}[f] - \mathbb{E}[f]| \geq \lim_{k \rightarrow \infty} \sup_{\nu \in K_{\tilde{\epsilon}_k, \bar{\pi}}(\mu)} |E_\nu[f] - \mathbb{E}[f]| - \delta \\ &= \lim_{\epsilon \rightarrow 0} \sup_{\nu \in K_{\epsilon, \bar{\pi}}(\mu)} |E_\nu[f] - \mathbb{E}[f]| - \delta, \end{aligned}$$

where the last inequality is because the sequence is decreasing. Letting $\delta \rightarrow 0$ completes the proof. The opposite inequality is obvious by the definition of $KL_{\pi}(\mu, \nu)$. \square

Proof of Theorem 2. For any bounded continuous $f : \Theta \rightarrow \mathbb{R}$, we have

$$\begin{aligned} & \left| E_g[f(\theta^u)|X^{\mathcal{U}}] - \mathbb{E}[f(\theta^u)|X^u] \right| \leq \|f\|_{\infty} g(K_{\epsilon, \pi}^C(\mu)|X^{\mathcal{U}}) \\ & + \int_{\nu \in K_{\epsilon, \pi}(\mu)} \left| E_{\nu}[f(\theta^u)|X^u] - \mathbb{E}[f(\theta^u)|X^u] \right| dg(\nu|X^{\mathcal{U}}), \end{aligned}$$

According to Lemma 3, the first term in the r.h.s. goes to 0 as $|\mathcal{U}| \rightarrow \infty$, μ -a.s. The second term can always be upper-bounded by

$$\sup_{\nu \in K_{\epsilon, \pi}(\mu)} \left| E_{\nu}[f(\theta^u)|X^u] - \mathbb{E}[f(\theta^u)|X^u] \right|.$$

By Bayes theorem,

$$E_{\nu}[f(\theta^u)|X^u] = \frac{E_{\nu}[f(\theta^u)P_{\theta^u}(X^u)]}{P_{\nu}(X^u)}.$$

For any $x \in \mathcal{H}_{\pi}$, Lemma 4 applied to the bounded continuous function $\theta \mapsto P_{\theta}(x)$ yields

$$\lim_{\epsilon \rightarrow 0} \sup_{\nu \in K_{\epsilon, \pi}(\mu)} \left| P_{\nu}(x) - \mathbb{P}(x) \right| = 0.$$

Another application of Lemma 4 to $\theta^u \mapsto \theta_{i,j}^u P_{\theta^u}(X^u)$ completes the proof. \square

Note that we could have obtained a version of the Theorem 2 giving a bound on the error in the estimation of any bounded continuous function $f(\theta^u)$ by simply using the function $f(\theta^u)P_{\theta^u}(X^u)$ in the last line of the above proof.

REFERENCES

- [1] A. J. Nicholson and B. D. Noble, “Breadcrumbs: forecasting mobile connectivity,” in *Proceedings of ACM MobiCom*, 2008.
- [2] V. A. Siris and D. Kalyvas, “Enhancing mobile data offloading with mobility prediction and prefetching,” *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 17, no. 1, pp. 22–29, 2013.
- [3] Y. Chon, H. Shin, E. Talipov, and H. Cha, “Evaluating mobility models for temporal prediction with high-granularity mobility data,” in *Pervasive Computing and Communications (PerCom)*. IEEE, 2012.
- [4] L. Song, D. Kotz, R. Jain, and X. He, “Evaluating next-cell predictors with extensive wi-fi mobility data,” *Mobile Computing, IEEE Transactions on*, vol. 5, no. 12, pp. 1633–1649, 2006.
- [5] S. Scellato, M. Musolesi, C. Mascolo, V. Latora, and A. T. Campbell, “Nextplace: a spatio-temporal prediction framework for pervasive systems,” in *Pervasive Computing*. Springer, 2011, pp. 152–169.
- [6] P. Jacquet, W. Szpankowski, and I. Apostol, “A universal predictor based on pattern matching,” *Information Theory, IEEE Transactions on*, vol. 48, no. 6, pp. 1462–1472, 2002.
- [7] T. S. Ferguson, “A bayesian analysis of some nonparametric problems,” *The annals of statistics*, pp. 209–230, 1973.
- [8] J. D. McAuliffe, D. M. Blei, and M. I. Jordan, “Nonparametric empirical bayes for the dirichlet process mixture model,” *Statistics and Computing*, vol. 16, no. 1, pp. 5–14, 2006.
- [9] N. Merhav, M. Feder, and M. Gutman, “Some properties of sequential predictors for binary markov sources,” *Information Theory, IEEE Transactions on*, vol. 39, no. 3, pp. 887–892, 1993.
- [10] P. Smyth *et al.*, “Clustering sequences with hidden markov models,” *Advances in neural information processing systems*, pp. 648–654, 1997.
- [11] T. Jebara, Y. Song, and K. Thadani, “Spectral clustering and embedding with hidden markov models,” in *Machine Learning: ECML 2007*. Springer, 2007, pp. 164–175.
- [12] J. McInerney, J. Zheng, A. Rogers, and N. R. Jennings, “Modelling heterogeneous location habits in human populations for location prediction under data sparsity,” in *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing*. ACM, 2013, pp. 469–478.
- [13] O. Kallenberg, *Foundations of modern probability*. springer, 2002.
- [14] D. Blackwell and J. B. MacQueen, “Ferguson distributions via pólya urn schemes,” *The annals of statistics*, pp. 353–355, 1973.
- [15] S. J. Gershman and D. M. Blei, “A tutorial on bayesian nonparametric models,” *Journal of Mathematical Psychology*, vol. 56, no. 1, 2012.
- [16] R. M. Neal, “Markov chain sampling methods for dirichlet process mixture models,” *Journal of computational and graphical statistics*, vol. 9, no. 2, pp. 249–265, 2000.
- [17] J. S. Liu, “Nonparametric hierarchical bayes via sequential imputations,” *The Annals of Statistics*, pp. 911–930, 1996.
- [18] S. Ghosal, J. K. Ghosh, and R. Ramamoorthi, “Posterior consistency of dirichlet mixtures in density estimation,” *Annals of Statistics*, vol. 27, no. 1, pp. 143–158, 1999.
- [19] S. Petrone, J. Rousseau, and C. Scricciolo, “Bayes and empirical bayes: do they merge?” *arXiv preprint arXiv:1204.1470*, 2012.
- [20] L. Schwartz, “On bayes procedures,” *Zeitschrift für Wahrscheinlichkeitstheorie und verwandte Gebiete*, vol. 4, no. 1, pp. 10–26, 1965.
- [21] Y. Chon, N. D. Lane, Y. Kim, F. Zhao, and H. Cha, “Understanding the coverage and scalability of place-centric crowdsensing,” in *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing*. ACM, 2013, pp. 3–12.
- [22] Y.-A. de Montjoye, Z. Smoreda, R. Trinquart, C. Ziemlicki, and V. D. Blondel, “D4d-senegal: The second mobile phone data for development challenge,” *arXiv preprint arXiv:1407.4885*, 2014.
- [23] S. J. Schiff, P. So, T. Chang, R. E. Burke, and T. Sauer, “Detecting dynamical interdependence and generalized synchrony through mutual prediction in a neural ensemble,” *Physical Review E*, vol. 54, 1996.
- [24] C. Villani, *Optimal transport: old and new*. Springer, 2008, vol. 338.